CS 188 Summer 2019 Introduction to Artificial Intelligence — Written HW 6 Sol.

Self-assessment due: Tuesday 8/13/2019 at 11:59pm (submit via Gradescope)

Q1. ML: Maximum Likelihood

Training Data

$(L = 1,\ F_1 = 1,\ F_2 = 1)$
$(L = 1,\ F_1 = 1,\ F_2 = 1)$
$(L = 0,\ F_1 = 1,\ F_2 = 1)$
$(L = 1,\ F_1 = 0,\ F_2 = 0)$
$(L = 0,\ F_1 = 0,\ F_2 = 0)$
$(L = 0,\ F_1 = 0,\ F_2 = 0)$
$(L = 0,\ F_1 = 0,\ F_2 = 1)$

You've decided to use a model-based approach to classification of text documents. Your goal is to build a classifier that can determine whether or not a document is about cats. You're taking a minimalist approach and you're only characterizing the input documents in terms of two binary features: F_1 and F_2. Both of these features have domain $\{0, 1\}$. The thing you're trying to predict is the label, L, which is also binary valued. When $L = 1$, the document is about cats. When $L = 0$, the document is not.

The particular meaning of the two features F_1 and F_2 is not important for your current purposes. You are only trying to decide on a particular Bayes' net structure for your classifier. You've got your hands on some training data (shown above) and you're trying to figure out which of several potential Bayes' nets (also shown above) might yield a decent classifier when trained on that training data.

(a) Which of the Bayes' nets, once learned from the training data with maximum likelihood estimation, would assign non-zero probability to the following query: $P(L = 1|F_1 = 0, F_2 = 0)$? Fill in all that apply.

● (i) ● (ii) ● (iii) ● (iv)

(b) Which of the Bayes' nets, once learned from the training data with maximum likelihood estimation, would assign non-zero probability to the following query: $P(L = 1|F_1 = 0, F_2 = 1)$? Fill in all that apply.

○ (i) ● (ii) ● (iii) ○ (iv)

(c) Which of the Bayes' nets, once learned from the training data with Laplace smoothing using $k = 1$, would assign non-zero probability to the following query: $P(L = 1|F_1 = 0, F_2 = 1)$? Fill in all that apply.

● (i) ● (ii) ● (iii) ● (iv)

(d) What probability does Bayes' net (i), once learned from the training data with Laplace smoothing using $k = 1$, assign to the query $P(L = 1|F_1 = 0, F_2 = 1)$?

$\frac{1}{3}$

(e) As $k \to \infty$ (the constant used for Laplace smoothing), what does the probability that Bayes' net (i) assigns to the query $P(L = 1|F_1 = 0, F_2 = 1)$ converge to?

$\frac{1}{2}$

Q2. Spam Classification

The Naïve Bayes model has been famously used for classifying spam. We will use it in the "bag-of-words" model:

- Each email has binary label Y which takes values in $\{\text{spam}, \text{ham}\}$.
- Each word w of an email, no matter where in the email it occurs, is assumed to have probability $P(W = w \mid Y)$, where W takes on words in a pre-determined dictionary. Punctuation is ignored.
- Take an email with K words $w_1, \ldots, w_K$. For instance: email "`hi hi you`" has $w_1 = \text{hi}, w_2 = \text{hi}, w_3 = \text{you}$. Its label is given by $\arg\max_y P(Y = y \mid w_1, \ldots, w_K) = \arg\max_y P(Y = y) \prod_{i=1}^{K} P(W = w_i \mid Y = y)$.

(a) You are in possession of a bag of words spam classifier trained on a large corpus of emails. Below is a table of some estimated word probabilities.

W	note	to	self	become	perfect
$P(W \mid Y = \text{spam})$	1/6	1/8	1/4	1/4	1/8
$P(W \mid Y = \text{ham})$	1/8	1/3	1/4	1/12	1/12

You are given a new email to classify, with only two words:

`perfect note`

Fill in the circles corresponding to all values of $P(Y = \text{spam})$ for which the bag of words with these word probabilities will give "spam" as the most likely label.

- ○ 0
- ○ 0.2
- ● 0.4
- ● 0.6
- ● 0.8
- ● 1

$$P(Y = \text{spam} \mid w_1 = \text{perfect}, w_2 = \text{note}) > P(Y = \text{ham} \mid w_1 = \text{perfect}, w_2 = \text{note})$$
$$P(w_1 = \text{perfect} \mid Y = \text{s})P(w_2 = \text{note} \mid Y = \text{s})P(Y = \text{s}) > P(w_1 = \text{perfect} \mid Y = \text{h})P(w_2 = \text{note} \mid Y = \text{h})P(Y = \text{h})$$
$$1/8 * 1/6 * P(Y = \text{spam}) > 1/12 * 1/8 * (1 - P(Y = \text{spam}))$$
$$2/96 * P(Y = \text{spam}) > 1/96 - 1/96 * P(Y = \text{spam})$$
$$3/96 * P(Y = \text{spam}) > 1/96$$
$$P(Y = \text{spam}) > 1/3$$

(b) You are given only three emails as a training set:

(Spam) `dear sir, I write to you in hope of recovering my gold watch.`
(Ham) `hey, lunch at 12?`
(Ham) `fine, watch it tomorrow night.`

Fill in the circles corresponding to values you would estimate for the given probabilities, if you were doing no smoothing.

$P(W = \textbf{sir} \mid Y = \textbf{spam})$	○ 0	○ 1/10	○ 1/5	○ 1/3	○ 2/3	● None of the above
$P(W = \textbf{watch} \mid Y = \textbf{ham})$	○ 0	○ 1/10	○ 1/5	○ 1/3	○ 2/3	● None of the above
$P(W = \textbf{gauntlet} \mid Y = \textbf{ham})$	● 0	○ 1/10	○ 1/5	○ 1/3	○ 2/3	○ None of the above
$P(Y = \textbf{ham})$	○ 0	○ 1/10	○ 1/5	○ 1/3	● 2/3	○ None of the above

Intuitively, the conditional probability is P(word | it's a word in an email of type Y). We estimate this probability with word counts:

$$P(W = \text{sir} \mid Y = \text{spam}) = \frac{c_w(W = \text{sir}, Y = \text{spam})/c_w(\text{total})}{c_w(Y = \text{spam})/c_w(\text{total})}$$
$$= \frac{c_w(W = \text{sir}, Y = \text{spam})}{c_w(Y = \text{spam}))} = \frac{1}{13}$$

Similarly, $P(W = \text{watch} \mid Y = \text{ham}) = \frac{1}{9}$.

The word "gauntlet" does not occur in our training emails, and since we're not smoothing, we estimate its conditional probability to be 0.

Estimating the prior probability $P(Y = \text{ham})$ only requires counting emails: $\frac{c_e(Y=\text{ham})}{c_e(\text{total})} = \frac{2}{3}$.

(c) You are training with the same emails as in the previous question, but now doing Laplace Smoothing with $k = 2$. There are V words in the dictionary. Write concise expressions for:

$P(W = \textbf{sir} \mid Y = \textbf{spam})$

"Sir" occurs once in the spam emails, and there are 13 total words in all spam emails. Smoothing with $k = 2$ gives $\frac{1+2}{13+2V}$.

$P(W = \textbf{watch} \mid Y = \textbf{ham})$

"Watch" occurs once in the ham emails, and there are nine total words in all ham emails. Smoothing with $k = 2$ gives $\frac{1+2}{9+2V}$.

$P(Y = \textbf{ham})$

Same as before: number of ham emails divided by the number of spam emails, which is 2/3. If you did Laplace smoothing on the priors as well, this would be 4/7, which was also accepted.

(d) On the held-out set, you see the following accuracies for different values of k:

k	0	1	2	10
accuracy	0.65	0.68	0.74	0.6

On the training set, the accuracy for $k = 0$ is 0.8. Fill in the circle of all plausible accuracies for $k = 10$ on the training set.

- ○ 0.1
- ● 0.7
- ○ 0.99
- ○ None of the above

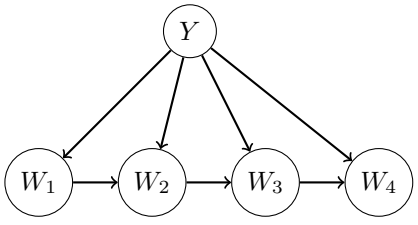
On the training set, smoothing counts is not likely to help as much as on the held-out set. Smoothing is supposed to prevent "overfitting" – but overfitting is what gives really high accuracy on the training set. There is also an optimal spot for smoothing parameter values, and we see that at $k = 10$, we went past that optimal spot on the held-out set. So we have a technique that is designed to prevent overfitting (in other words, lower accuracy) on the training set, and it's so strong that it's actually hurting performance even on the held-out set: it definitely shouldn't be expected to increase performance on the training set! This rules out 0.99 as an answer.

Why is 0.1 ruled out? Because achieving a classification error of 0.1 is equivalent to achieving classification *accuracy* of 0.9 – we have to flip the sign, but the signal is there. But that would mean that we improved performance from accuracy of 0.8, which we decided above is not reasonable to expect.

0.7, however, is a reasonable answer: it hurts performance compared to not doing smoothing, which is what we should expect.

(e) Becoming less naïve

We are now going to improve the representational capacity of the model. Presence of word w_i will be modeled not by $P(W = w_i \mid Y)$, where it is only dependent on the label, but by $P(W = w_i \mid Y, W_{i-1})$, where it is also dependent on the previous word. The corresponding model for an email of only four words is given on the right.



(i) With a vocabulary consisting of V words, what is the *minimal* number of conditional **word** probabilities that need to be estimated for this model? The correct answer is among the choices.

- ○ V
- ○ $2V$
- ○ V^2
- ● $2V^2$
- ○ 2^V
- ○ 2^{2V}

We need to consider V possible words for each one of V possible previous words and 2 possible lables: $2V^2$.

(ii) Select all expected effects of using the new model instead of the old one, if both are trained with a very large set of emails (equal number of spam and ham examples).

- ◯ The entropy of the posterior $P(Y|W)$ should on average be lower with the new model. (In other words, the model will tend to be more confident in its answers.)
- ● The accuracy on the **training** data should be higher with the new model.
- ● The accuracy on the **held-out** data should be higher with the new model.
- ◯ None of the above.

The new model is closer to an actual model of language, and so should better model emails and thus filter spam from non-spam on both the training and held-out datasets. Remember that Naïve Bayes is an *overconfident* model: it gives unwarrantedly low-entropy posteriors. This model is less "naïve", and is therefore less overconfident — its posterior can be expected to be higher entropy.

Q3. Potpourri

(a) A single perceptron can compute the XOR function.

○ True ● False

(b) A perceptron is guaranteed to learn a separating decision boundary for a separable dataset within a finite number of training steps.

● True ○ False

(c) Given a linearly separable dataset, the perceptron algorithm is guaranteed to find a max-margin separating hyperplane.

○ True ● False

(d) You would like to train a neural network to classify digits. Your network takes as input an image and outputs probabilities for each of the 10 classes, 0-9. The network's prediction is the class that it assigns the highest probability to. From the following functions, select all that would be suitable loss functions to minimize using gradient descent:

☐ The square of the difference between the correct digit and the digit predicted by your network

☐ The probability of the correct digit under your network

■ The negative log-probability of the correct digit under your network

○ None of the above

- Option 1 is incorrect because it is non-differentiable. The correct digit and your model's predicted digit are both integers, and the square of their difference takes on values from the set $\{0^2, 1^2, \ldots, 9^2\}$. Losses that can be used with gradient descent must take on values from a continuous range and have well-defined gradients.
- Option 2 is not a loss because you would like to *maximize* the probability of the correct digit under your model, not minimize it.
- Option 3 is a common loss used for classification tasks. When the probabilities produced by a neural network come from a softmax layer, this loss is often combined with the softmax computation into a single entity known as the "softmax loss" or "softmax cross-entropy loss".

Q4. Perceptron and Kernels

A kernel is a mapping $K(x, y)$ from pairs vectors in $\mathbb{R}^d$ into the real numbers such that $K(x, y) = \Phi(x) \cdot \Phi(y)$ where Φ is a mapping from $\mathbb{R}^d$ into $\mathbb{R}^D$ where D is possibly different from d and even infinite. We say that a mapping $K(x, y)$ for which such Φ exists is a valid kernel.

(a) The following binary class data has two features, A and B.

Index	A	B	Class
1.	1	1	1
2.	0	3	-1
3.	1	-1	1
4.	3	0	-1
5.	-1	1	1
6.	0	-3	-1
7.	-1	-1	1
8.	-3	0	-1

(i) Select all true statements:

- ☐ This data is linearly separable.
- ■ This data is linearly separable if we use a feature map $\phi((A, B)) = (A^2, B^2, 1)$.
- ■ There exists a kernel such that this data is linearly separable.
- ■ For all datasets in which no data point is labeled in more than one distinct way, there exists a kernel such that the data is linearly separable.
- ☐ For all datasets, there exists a kernel such that the data is linearly separable.
- ☐ For all valid kernels, there exists a dataset with at least one point from each class that is linearly separable under that kernel.
- ☐ None of the above.

We will be running both the primal (normal) binary (not multiclass) perceptron and dual binary perceptron algorithms on this dataset. We will initialize the weight vector w to $(1, 1)$ for the primal perceptron algorithm. Accordingly, we will initialize the α vector to $(1, 0, 0, 0, 0, 0, 0, 0)$ for the dual perceptron algorithm with the kernel $K(x, y) = x \cdot y$. Pass through the data using the indexing order provided. There is no bias term.

Write your answer in the box provided. Show your work outside of the boxes to have a chance at receiving partial credit.

(ii) What is the first misclassified point?

Point 2.

(iii) For the *primal* perceptron algorithm, what is the weight vector after the first weight update?

The weight vector after the first weight update will be:

$$w = (1, 1) - (0, 3) = (1, -2) \quad (1)$$

For your convenience, the data is duplicated on this page.

Index	A	B	Class
1.	1	1	1
2.	0	3	-1
3.	1	-1	1
4.	3	0	-1
5.	-1	1	1
6.	0	-3	-1
7.	-1	-1	1
8.	-3	0	-1

(iv) For the *dual* perceptron algorithm, what is the α vector after the first weight update?

The α vector after the first update will be:

$$\alpha = (1, -1, 0, 0, 0, 0, 0, 0) \quad (2)$$

(v) What is the second misclassified point?

Point 4.

(vi) For the *primal* perceptron algorithm, what is the weight vector after the second weight update?

The weights after the second weight update will be:

$$w = (1, -2) - (3, 0) = (-2, -2) \quad (3)$$

(vii) For the *dual* perceptron algorithm, what is the α vector after the second weight update?

The α vector after the second update will be:

$$\alpha = (1, -1, 0, -1, 0, 0, 0, 0) \quad (4)$$

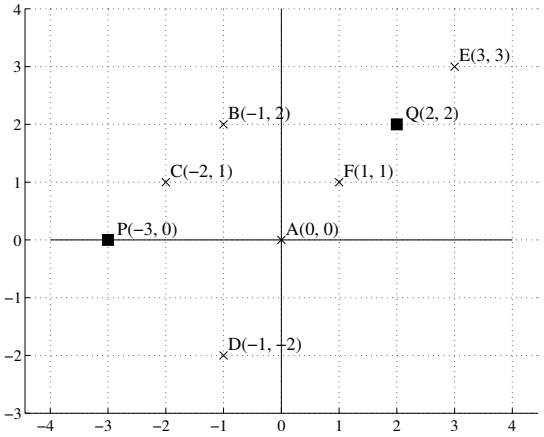
(b) Consider the following kernel function: $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^2$ where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$. Find a valid Φ map for this kernel. That is, find a vector-to-vector function ϕ such that $\phi(\mathbf{x}) \cdot \phi(\mathbf{y}) = K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^2$. Show work to have a chance at receiving partial credit. Any precise answer format is acceptable.

Expanding $(x \cdot y)^2 = (x_1 y_1 + x_2 y_2)^2 = x_1^2 y_1^2 + 2 x_1 y_1 x_2 y_2 + x_2^2 y_2^2$ so the mapping $\Phi(x) = [x_1^2, \sqrt{2} x_1 x_2, x_2^2]$ is valid.

Q5. Clustering

In this question, we will do k-means clustering to cluster the points $A, B \ldots F$ (indicated by $\times$'s in the figure on the right) into 2 clusters. The current cluster centers are P and Q (indicated by the ■ in the diagram on the right). Recall that k-means requires a distance function. Given 2 points, $A = (A_1, A_2)$ and $B = (B_1, B_2)$, we use the following distance function $d(A, B)$ that you saw from class,

$$d(A, B) = (A_1 - B_1)^2 + (A_2 - B_2)^2$$



(a) Update assignment step: Select all points that get assigned to the cluster with center at P:

○ A ● B ● C ● D ○ E ○ F ○ No point gets assigned to cluster P

(b) Update cluster center step: What does cluster center P get updated to?

The cluster center gets updated to the point, P' which minimizes, $d(P', B) + d(P', C) + d(P', D)$, which in this case turns out to be the centroid of the points, hence the new cluster center is

$$\left(\frac{-1-2-1}{3}, \frac{2+1-2}{3}\right) = \left(\frac{-4}{3}, \frac{+1}{3}\right)$$

Changing the distance function: While k-means used Euclidean distance in class, we can extend it to other distance functions, where the assignment and update phases still iteratively minimize the total (non-Euclidian) distance. Here, consider the Manhattan distance:

$$d'(A, B) = |A_1 - B_1| + |A_2 - B_2|$$

We again start from the original locations for P and Q as shown in the figure, and do the update assignment step and the update cluster center step using Manhattan distance as the distance function:

(c) Update assignment step: Select all points that get assigned to the cluster with center at P, under this new distance function $d'(A, B)$.

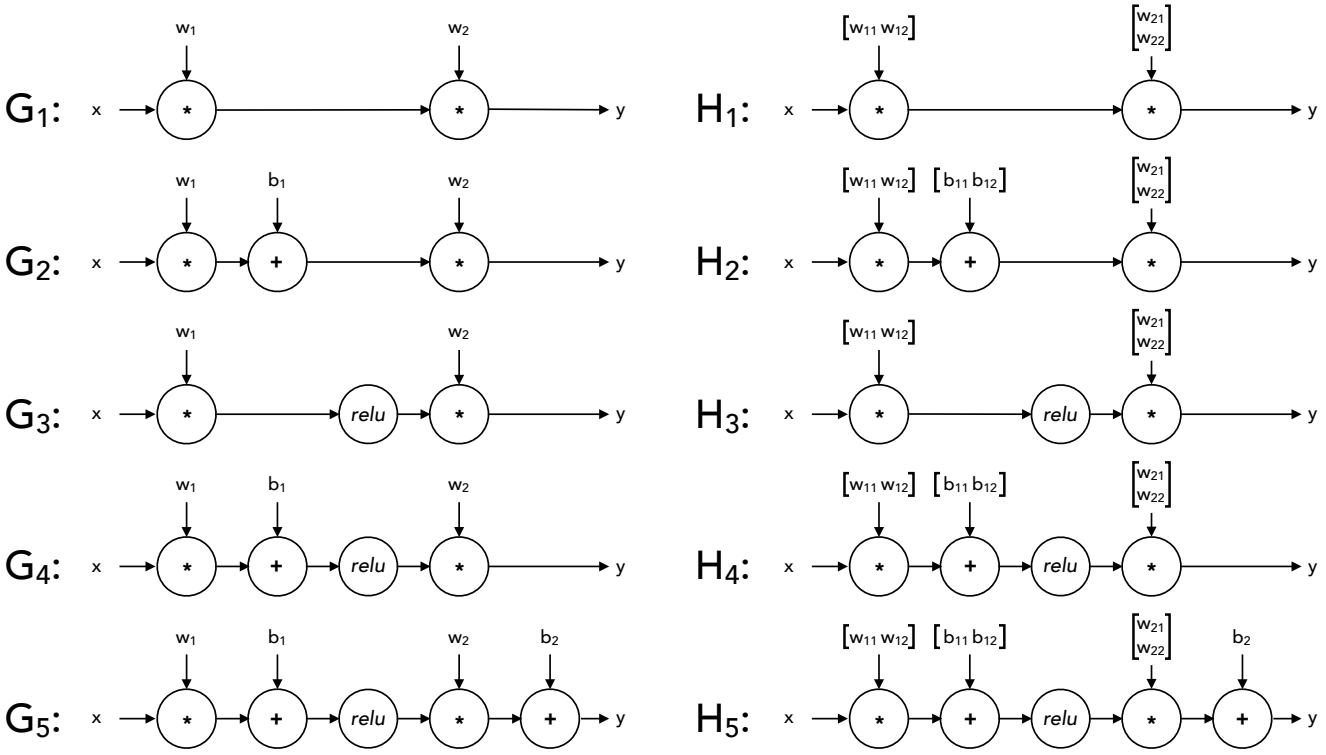
● A ○ B ● C ● D ○ E ○ F ○ No point gets assigned to cluster P

(d) Update cluster center step: What does cluster center P get updated to, under this new distance function $d'(A, B)$?

The cluster center gets updated to the point, P' which minimizes, $d'(P', A) + d'(P', C) + d'(P', D)$, which in this case turns out to be the point with X-coordinate as the median of the X-coordinate of the points in the cluster and the Y-coordinate as the median of the Y-coordinate of the points in the cluster. Hence the new cluster center is

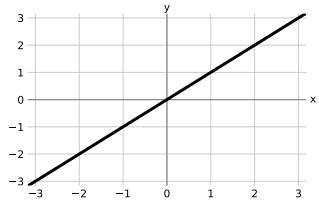
$$(-1, 0)$$

Q6. Neural Networks: Representation

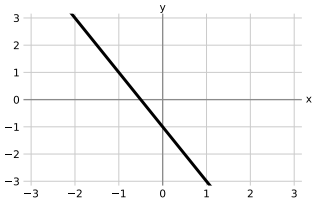


For each of the piecewise-linear functions below, mark all networks from the list above that can represent the function **exactly** on the range $x \in (-\infty, \infty)$. In the networks above, *relu* denotes the element-wise ReLU nonlinearity: $relu(z) = max(0, z)$. The networks G_i use 1-dimensional layers, while the networks H_i have some 2-dimensional intermediate layers.

(a)



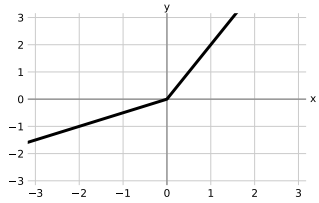
- [x] G_1 [x] H_1
- [x] G_2 [x] H_2
- [ ] G_3 [ ] H_3
- [ ] G_4 [x] H_4
- [ ] G_5 [x] H_5
- ( ) None of the above



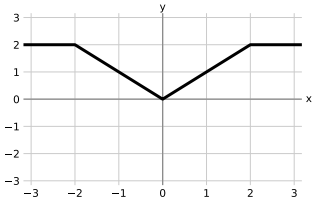
- [ ] G_1 [ ] H_1
- [x] G_2 [x] H_2
- [ ] G_3 [ ] H_3
- [ ] G_4 [x] H_4
- [ ] G_5 [x] H_5
- ( ) None of the above

The networks G_3, G_4, G_5 include a ReLU nonlinearity on a scalar quantity, so it is impossible for their output to represent a non-horizontal straight line. On the other hand, H_3, H_4, H_5 have a 2-dimensional hidden layer, which allows two ReLU elements facing in opposite directions to be added together to form a straight line. The second subpart requires a bias term because the line does not pass through the origin.

(b)



- [ ] G_1 [ ] H_1
- [ ] G_2 [ ] H_2
- [ ] G_3 [x] H_3
- [ ] G_4 [x] H_4
- [ ] G_5 [x] H_5
- ( ) None of the above



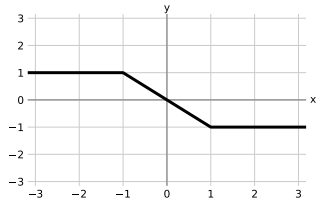
- [ ] G_1 [ ] H_1
- [ ] G_2 [ ] H_2
- [ ] G_3 [ ] H_3
- [ ] G_4 [ ] H_4
- [ ] G_5 [ ] H_5
- (●) None of the above

These functions include multiple non-horizontal linear regions, so they cannot be represented by any of the networks G_i which apply ReLU no more than once to a scalar quantity.

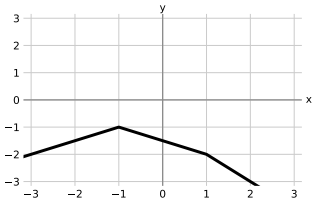
The first subpart can be represented by any of the networks with 2-dimensional ReLU nodes. The point of nonlinearity occurs at the origin, so nonzero bias terms are not required.

The second subpart has 3 points where the slope changes, but the networks H_i only have a single 2-dimensional ReLU node. Each application of ReLU to one element can only introduce a change of slope for a single value of x.

(c)



☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ☐ H_4
☐ G_5 ■ H_5
◯ None of the above



☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ■ H_4
☐ G_5 ■ H_5
◯ None of the above

Both functions have two points where the slope changes, so none of the networks $G_i; H_1, H_2$ can represent them.

An output bias term is required for the first subpart because one of the flat regions must be generated by the flat part of a ReLU function, but neither one of them is at $y = 0$.

The second subpart doesn't require a bias term at the output: it can be represented as $-relu(\frac{-x+1}{2}) - relu(x+1)$. Note how if the segment at $x > 2$ were to be extended to cross the x axis, it would cross exactly at $x = -1$, the location of the other slope change. A similar statement is true for the segment at $x < -1$.

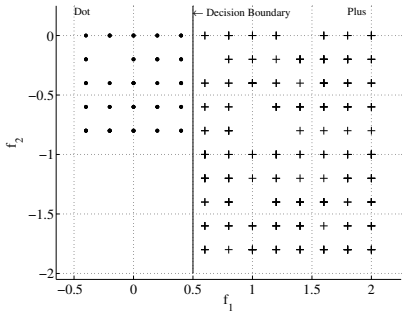
Q7. Decision Trees

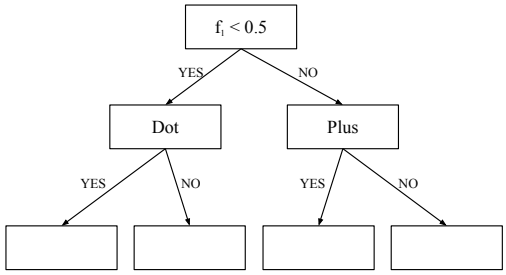
You are given points from 2 classes, shown as +'s and ·'s. For each of the following sets of points,

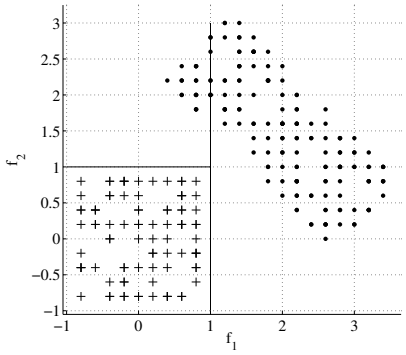
1. Draw the decision tree of depth at most 2 that can separate the given data completely, by filling in binary predicates (which only involve thresholding of a *single* variable) in the boxes for the decision trees below. If the data is already separated when you hit a box, simply write the class, and leave the sub-tree hanging from that box empty.

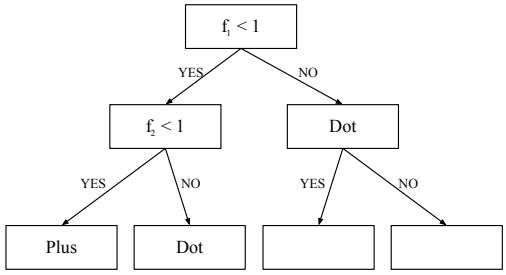
2. Draw the corresponding decision boundaries on the scatter plot, and write the class labels for each of the resulting bins somewhere inside the resulting bins.

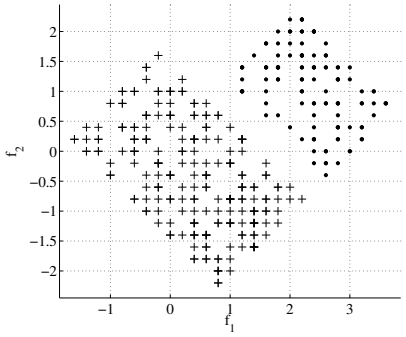
If the data can not be separated completely by a depth 2 decision tree, simply cross out the tree template. We solve the first part as an example.

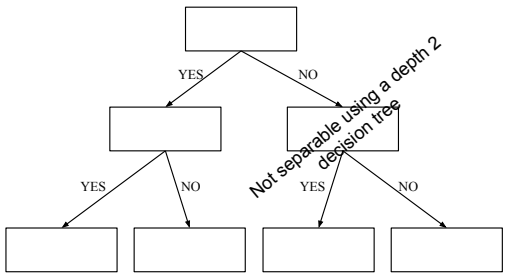


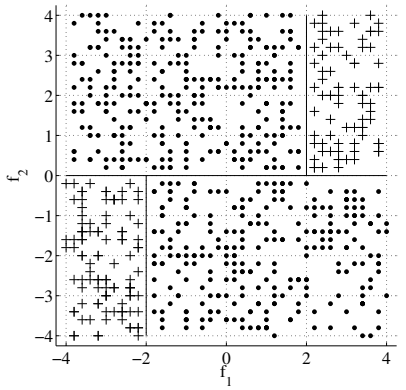


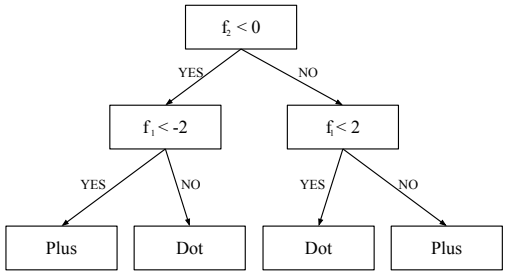








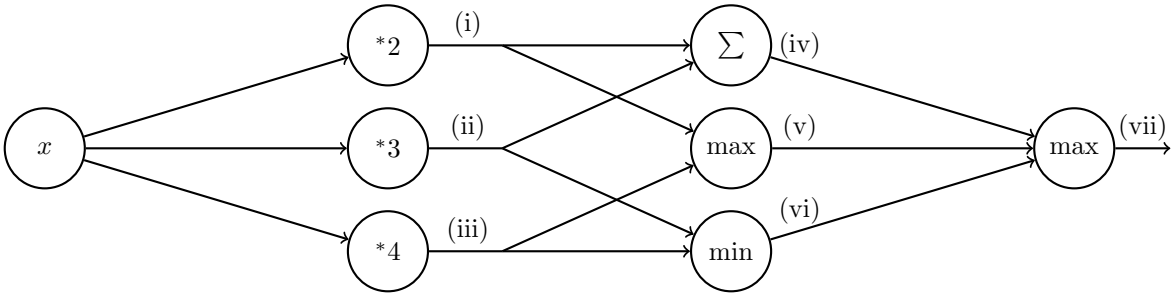




Q8. Deep Learning

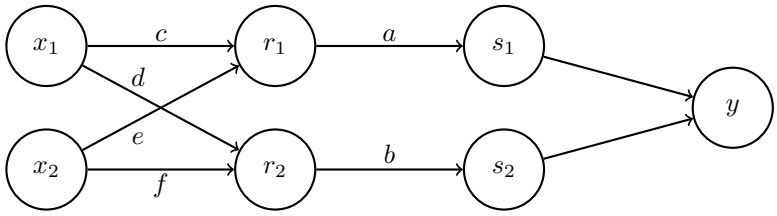
(a) Perform forward propagation on the neural network below for $x = 1$ by filling in the values in the table. Note that (i), ..., (vii) are outputs after performing the appropriate operation as indicated in the node.

(i)	(ii)	(iii)	(iv)	(v)	(vi)	(vii)
2	3	4	5	4	3	5



(b) Below is a neural network with weights a, b, c, d, e, f. The inputs are x_1 and x_2.
The first hidden layer computes $r_1 = \max(c \cdot x_1 + e \cdot x_2, 0)$ and $r_2 = \max(d \cdot x_1 + f \cdot x_2, 0)$.
The second hidden layer computes $s_1 = \frac{1}{1+\exp(-a \cdot r_1)}$ and $s_2 = \frac{1}{1+\exp(-b \cdot r_2)}$.
The output layer computes $y = s_1 + s_2$. Note that the weights a, b, c, d, e, f are indicated along the edges of the neural network here.

Suppose the network has inputs $x_1 = 1, x_2 = -1$.
The weight values are $a = 1, b = 1, c = 4, d = 1, e = 2, f = 2$.
Forward propagation then computes $r_1 = 2, r_2 = 0, s_1 = 0.9, s_2 = 0.5, y = 1.4$. Note: some values are rounded.



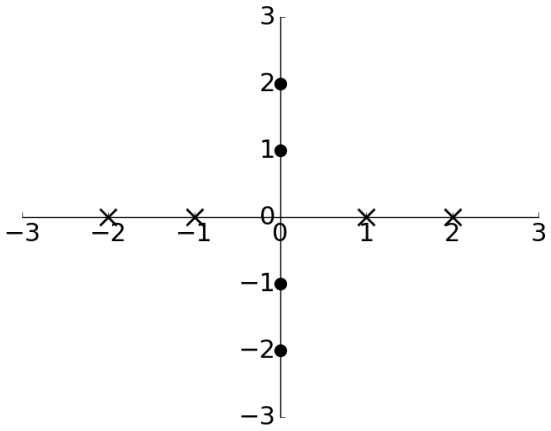
Using the values computed from forward propagation, use backpropagation to numerically calculate the following partial derivatives. Write your answers as a single number (not an expression). You do not need a calculator. Use scratch paper if needed.
Hint: For $g(z) = \frac{1}{1+\exp(-z)}$, the derivative is $\frac{\partial g}{\partial z} = g(z)(1 - g(z))$.

$\frac{\partial y}{\partial a}$	$\frac{\partial y}{\partial b}$	$\frac{\partial y}{\partial c}$	$\frac{\partial y}{\partial d}$	$\frac{\partial y}{\partial e}$	$\frac{\partial y}{\partial f}$
0.18	0	0.09	0	−0.09	0

$$
\begin{aligned}
\frac{\partial y}{\partial a} &= \frac{\partial y}{\partial s_1}\frac{\partial s_1}{\partial a} \\
&= 1 \cdot \frac{\partial g(a \cdot r_1)}{\partial a} \\
&= r_1 \cdot g(a \cdot r_1)(1 - g(a \cdot r_1)) \\
&= r_1 \cdot s_1(1 - s_1) \\
&= 2 \cdot 0.9 \cdot (1 - 0.9) \\
&= 0.18 \\
\frac{\partial y}{\partial b} &= \frac{\partial y}{\partial s_2}\frac{\partial s_2}{\partial b} \\
&= 1 \cdot \frac{\partial g(b \cdot r_2)}{\partial b} \\
&= r_2 \cdot g(b \cdot r_2)(1 - g(b \cdot r_2)) \\
&= r_2 \cdot s_2(1 - s_2) \\
&= 0 \cdot 0.5(1 - 0.5) \\
&= 0 \\
\frac{\partial y}{\partial c} &= \frac{\partial y}{\partial s_1}\frac{\partial s_1}{\partial r_1}\frac{\partial r_1}{\partial c} \\
&= 1 \cdot [a \cdot g(a \cdot r_1)(1 - g(a \cdot r_1))] \cdot x_1 \\
&= [a \cdot s_1(1 - s_1)] \cdot x_1 \\
&= [1 \cdot 0.9(1 - 0.9)] \cdot 1 \\
&= 0.09 \\
\frac{\partial y}{\partial d} &= \frac{\partial y}{\partial s_2}\frac{\partial s_2}{\partial r_2}\frac{\partial r_2}{\partial d} \\
&= \frac{\partial y}{\partial s_2}\frac{\partial s_2}{\partial r_2} \cdot 0 \\
&= 0 \\
\frac{\partial y}{\partial e} &= \frac{\partial y}{\partial s_1}\frac{\partial s_1}{\partial r_1}\frac{\partial r_1}{\partial e} \\
&= 1 \cdot [a \cdot g(a \cdot r_1)(1 - g(a \cdot r_1))] \cdot x_2 \\
&= [a \cdot s_1(1 - s_1)] \cdot x_2 \\
&= [1 \cdot 0.9(1 - 0.9)] \cdot -1 \\
&= -0.09 \\
\frac{\partial y}{\partial f} &= \frac{\partial y}{\partial s_2}\frac{\partial s_2}{\partial r_2}\frac{\partial r_2}{\partial f} \\
&= \frac{\partial y}{\partial s_2}\frac{\partial s_2}{\partial r_2} \cdot 0 \\
&= 0
\end{aligned}
$$

(c) Below are two plots with horizontal axis x_1 and vertical axis x_2 containing data labelled $\times$ and $\bullet$. For each plot, we wish to find a function $f(x_1, x_2)$ such that $f(x_1, x_2) \geq 0$ for all data labelled $\times$ and $f(x_1, x_2) < 0$ for all data labelled $\bullet$.

Below each plot is the function $f(x_1, x_2)$ for that specific plot. Complete the expressions such that all the data is labelled correctly. If not possible, mark "No valid combination".



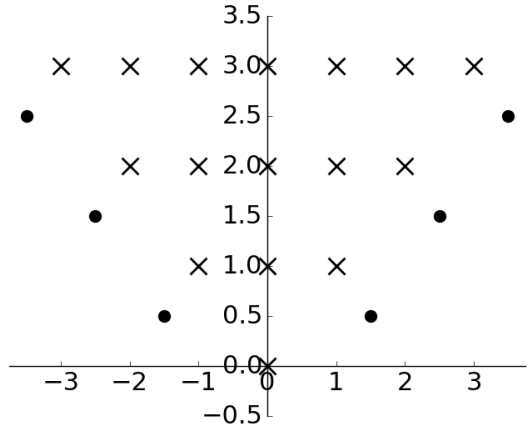
$f(x_1, x_2) = \max(\ \underline{\textbf{(i)}} + \underline{\textbf{(ii)}},\ \underline{\textbf{(iii)}} + \underline{\textbf{(iv)}}\ ) + \underline{\textbf{(v)}}$

(i)	● x_1	○ $-x_1$	○ 0
(ii)	○ x_2	○ $-x_2$	● 0
(iii)	○ x_1	● $-x_1$	○ 0
(iv)	○ x_2	○ $-x_2$	● 0
(v)	○ 1	● -1	○ 0

○ No valid combination

There are two possible solutions:

$$f(x_1, x_2) = \max(x_1, -x_1) - 1$$
$$f(x_1, x_2) = \max(-x_1, x_1) - 1$$



$f(x_1, x_2) = \underline{\textbf{(vi)}} - \max(\ \underline{\textbf{(vii)}} + \underline{\textbf{(viii)}},\ \underline{\textbf{(ix)}} + \underline{\textbf{(x)}}\ )$

(vi)	● x_2	○ $-x_2$	○ 0
(vii)	● x_1	○ $-x_1$	○ 0
(viii)	○ x_2	○ $-x_2$	● 0
(ix)	○ x_1	● $-x_1$	○ 0
(x)	○ x_2	○ $-x_2$	● 0

○ No valid combination

There are four possible solutions:

$$f(x_1, x_2) = x_2 - \max(x_1, -x_1)$$
$$f(x_1, x_2) = x_2 - \max(-x_1, x_1)$$
$$f(x_1, x_2) = -\max(x_1 - x_2, -x_1 - x_2)$$
$$f(x_2, x_2) = -\max(-x_1 - x_2, x_1 - x_2)$$